CPU-8E v2.0
Nova Interface Gráfica Multi-plataforma
Relatório do Projeto Final de AOC
Ciência da Computação
IESB – Instituto de Educação Superior de Brasília
3/12/2009

Por Bárbara Alves Bezerra dos Anjos - 0822130046
e Wanderlan Santos dos Anjos – 0822130045

Prof. Joel Guilherme

1. Objetivos

Dotar a CPU-8E de uma interface gráfica (GUI – Graphical User Interface) moderna, intuitiva e multi-plataforma. Essa nova interface deverá levar os alunos a terem uma nova experiência de aprendizado na arquitetura de computadores. Permitindo visualizar a interação entre cada componente da CPU, principalmente entre os registradores, os barramentos e a memória, através de elementos gráficos desenhados na tela. Além disso, a nova CPU-8E deverá funcionar com os mesmos recursos em Linux e Mac OS.

2. Motivação

A CPU-8E até a versão v.1.0b possuía uma interface caractere, que embora funcionasse adequadamente em Windows não apresentava sua janela console de forma correta no Linux. Este projeto além de resolver este problema tornou a CPU-8E muito mais didática, estendendo e consolidando os objetivos originais da CPU-8E criada pelo Prof. Joel.

3. Ferramenta Utilizada

Como a CPU-8E v.1.0 foi programada em FreePascal a escolha natural para implementar a versão 2.0 recaiu sobre a ferramenta de desenvolvimento "**Lazarus**"(www.lazarus.freepascal.org) .

Lazarus é um ambiente de desenvolvimento (IDE) OpenSource (GPL), totalmente construída em FreePascal para criar aplicativos em FreePascal. Ele suporta o paradigma RAD (Rapid Application Development), que facilita grandemente o desenvolvimento de aplicações com interface gráfica em praticamente qualquer plataforma existente no mercado. Lazarus é bastante similar ao Delphi de forma que muitos programadores terão facilidade em utilizá-lo.

Lazarus pode criar aplicações GUI para as seguintes plataformas:

1- Windows 32 ou 64 bits com GDI (a biblioteca nativa de widgets do Windows)
2- Linux 32 ou 64 bits com GTK2 (biblioteca nativa de widgets do Gnome) ou QT (biblioteca nativa de widgets do KDE).
3- FreeBSD com GTK2 (biblioteca nativa de widgets do Gnome) ou QT (biblioteca nativa de widgets do KDE).
4- MacOS com Carbon (biblioteca nativa de widgets da Apple).
5- Windows CE.

O mesmo código fonte da nova CPU-8E pode ser compilado para essas plataformas, usando Lazarus.

Para mais informações sobre Lazarus veja os links abaixo:

- http://www.lazarus.freepascal.org/index.php?PHPSESSID=6385a7237dd66d81b132f8c3a6335817&page=7
- http://www.linuxjournal.com/article/10502

4. Modelo da Interface

Partindo das especificações originais constantes do documento "CPU-8E - Specs - v1.0b" de autoria do Prof. Joel, notamos que a melhor base para a nova interface seria o próprio **Diagrama de Blocos** da CPU-8E apresentado no item 4 do referido documento e reproduzido na figura 1 abaixo.

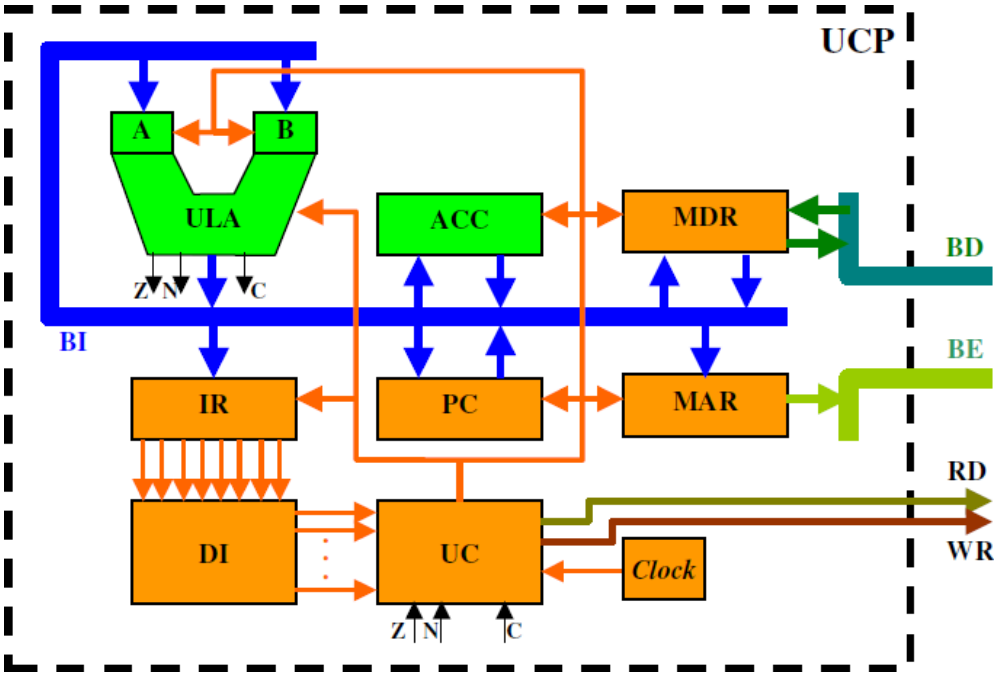

Figura 1 – Diagrama de Blocos original da CPU-8E

O diagrama foi então convertido no novo painel de controle da CPU-8E com alguns recursos de animação. Com isso o aluno pode executar os comandos e ver os barramentos e os blocos se acenderem dinamicamente para mostrar como exatamente uma instrução afeta os componentes da CPU. Também acrescentamos hints (dicas ou mini-helps), em cada elemento visual, que foram extraídos da própria especificação tornando o aprendizado mais fácil e direto.

A figura 2 mostra como ficou a nova interface.

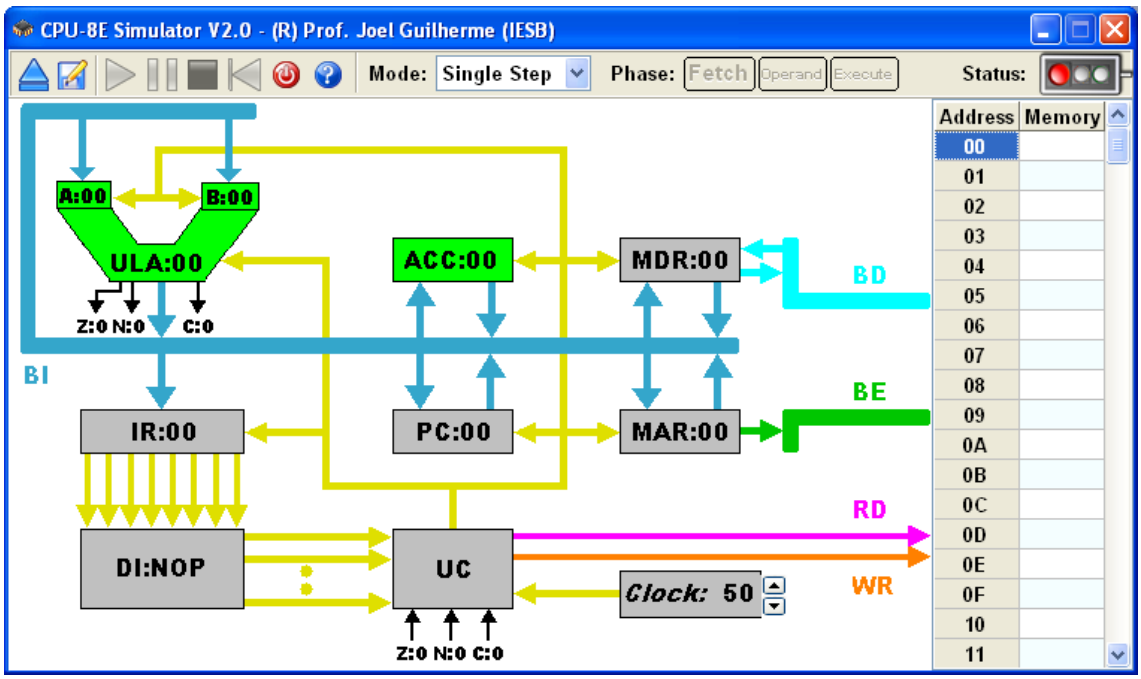

Figura 2 Interface convertida usando Lazarus rodando em Windows XP

Para detalhamento da operação da nova CPU leia o documento de ajuda "CPU8E_Sim.txt" que acompanha o executável do projeto.

5. Desenvolvimento Colaborativo

Também estabelecemos uma nova forma para o desenvolvimento da CPU-8E. Criamos com autorização do Prof. Joel um site no GoogleCode (http://cpu-8e.googlecode.com/). Com isso ficam facilitados e assegurados a divulgação e o desenvolvimento do projeto num site central. Outros alunos ou colaboradores poderão usar e participar do projeto bem como propor melhorias.

O GoogleCode é similar ao SourceForge porém muito mais fácil de usar, possui recursos de desenvolvimento colaborativo on-line, via Internet, para sistemas open-source com:

1- Subversion (controle de versão dos fontes e documentação).
2- Download de arquivos do projeto: fontes e executáveis.
3- Páginas wiki para documentação a quente
4- Bugtracking e controle de demandas.
5- Links.
6- Clara apresentação da licença open-source, e
7- Fórum de discussões.

Para mais detalhes sobre a utilização do GoogleCode veja:
http://code.google.com/p/support/wiki/GettingStarted

Para download dos arquivos e para desenvolvimento colaborativo do projeto CPU-8E acesse:
http://cpu-8e.googlecode.com/

6. Compilando o Projeto

Para compilar a CPU-8E em Windows você precisará realizar alguns procedimentos:

1- Baixe e instale Lazarus Windows 32 bits, versão 0.9.28.2 ou superior: http://sourceforge.net/projects/lazarus/files/

2- Baixe os fontes da CPU-8E: http://cpu-8e.googlecode.com/ e descomprima o pacote num diretório de trabalho

3- Aplicar, manualmente, um patch no Lazarus para acrescentar o desenho de triângulos no componente TShape:

1. Chame o Lazarus e abra o arquivo **c:\Lazarus\lcl\extctrls.pp**
2. Tecle <Ctrl+F> e ache o tipo "**TShape**"
3. Substitua a declaração existente por esta:

   TShapeType = (stRectangle, stSquare, stRoundRect, stRoundSquare, stEllipse, stCircle, stSquaredDiamond, stDiamond, stRightTriangle, stLeftTriangle, stUpTriangle, stDownTriangle);

4. Isso irá criar novas enumerações para TShape, no caso triângulos
5. Salve o arquivo.
6. Abra o arquivo **c:\Lazarus\lcl\include\shape.inc**
7. Tecle <Ctrl+F> e ache o método "**procedure TShape.Paint;**"
   Acrescente a variável local: *T: array[0..2] of TPoint;*
8. Acrescente no "**case FShape of**" o código:

```
stRightTriangle:
begin
  with Self do
  begin
    T[0].x := 0;
    T[0].y := 0;
    T[1].x := Width - 1;
    T[1].y := (Height - 1) div 2;
    T[2].x := 0;
    T[2].y := Height - 1;
    Polygon(T);
  end;
end;
stLeftTriangle:
begin
  with Self do
  begin
    T[0].x := 0;
    T[0].y := (Height - 1) div 2;
    T[1].x := Width - 1;
    T[1].y := 0;
    T[2].x := Width - 1;
    T[2].y := Height - 1;
```

```
    Polygon(T);
  end;
end;
stUpTriangle:
begin
  with Self do
  begin
    T[0].x := 0;
    T[0].y := Height - 1;
    T[1].x := (Width - 1) div 2;
    T[1].y := 0;
    T[2].x := Width - 1;
    T[2].y := Height - 1;
    Polygon(T);
  end;
end;
stDownTriangle:
begin
  with Self do
  begin
    T[0].x := 0;
    T[0].y := 0;
    T[1].x := Width - 1;
    T[1].y := 0;
    T[2].x := (Width - 1) div 2;
    T[2].y := Height - 1;
    Polygon(T);
  end;
end;
```

9. Salve o arquivo.
10. Reconstrua o Lazarus:
    a. Acesse no Lazarus o menu: Tools/Configure “Build Lazarus”.
    b. Em Quick Build Options click no radio-button: “Clean + Build all”.
    c. Click no botão Build
    d. Espere reentrada do Lazarus.
11. Abra o arquivo CPU8E_main.lpr, que está no diretório onde você copiou o projeto.
12. Comece a fazer suas alterações e implementações usando o Lazarus.

7. Conclusão

Observamos que a modificação da interface caractere para GUI foi muito facilitada pelo Lazarus com seu paradigma RAD orientado a eventos e a linguagem Object Pascal que, segundo nossa opinião, torna os projetos mais fáceis de entender e manter. Por fim o executável final não tem dependências de outras bibliotecas ou frameworks tornando a instalação da CPU-8E um mero procedimento de cópia de arquivos, ao contrário do que teríamos se tivéssemos optado por outros ambientes de desenvolvimento para realizarmos o remake da CPU-8E.